Cornell University CS 569: Interactive Computer Graphics

Soft shadows

Lecture 9

Soft Shadows: Heckbert & Herf

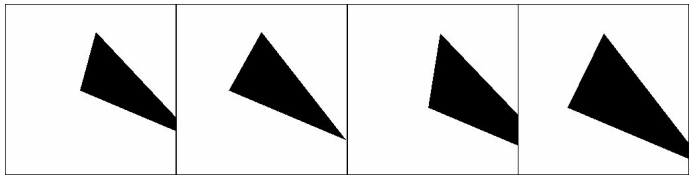
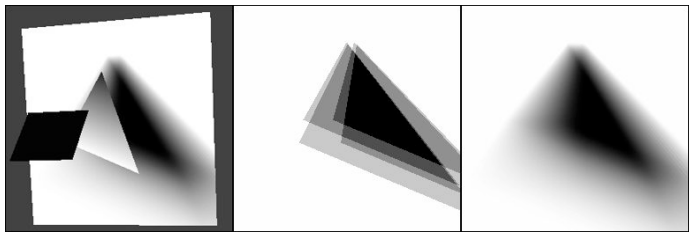


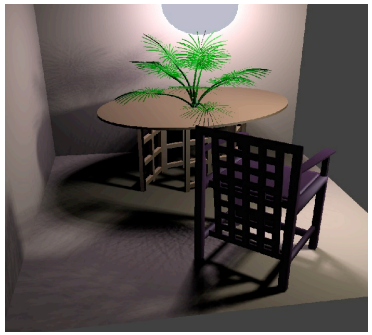
Figure 1: Hard shadow images from 2×2 grid of sample points on light source.



[Michael Herf and Paul Heckbert]

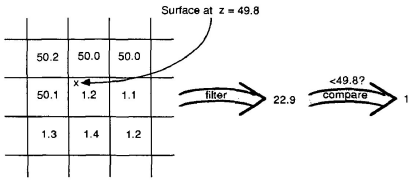
Figure 2: Left: scene with square light source (foreground), triangular occluder (center), and rectangular receiver (background), with shadows on receiver. Center: Approximate soft shadows resulting from 2×2 grid of sample points; the average of the four hard shadow images in Figure 1. Right: Correct soft shadow image (generated with 16×16 sampling). This image is used as the texture on the receiver at left.

Heckbert & Herf Soft Shadows

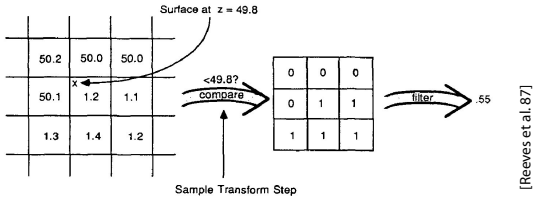


[Michael Herf and Paul Heckbert]

Percentage closer filtering



a) Ordinary texture map filtering. Does not work for depth maps.



[Reeves et al. 87]

b) Percentage closer filtering.

Figure 2. Ordinary filtering versus percentage closer filtering.

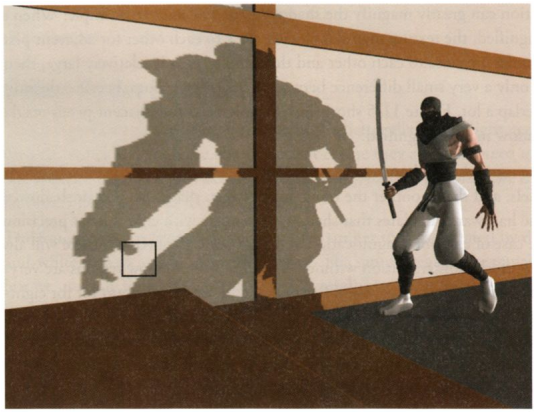


Figure 11-6. Ninja Shadow with One Sample per Pixel
Ninja model by William "Proton" Vaughn.

[Bunnell & Pellacini, *GPU Gems*]

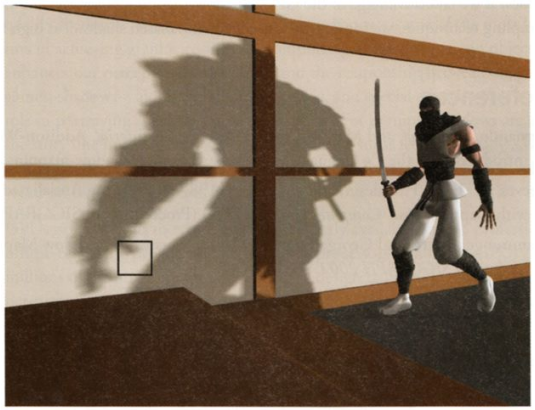
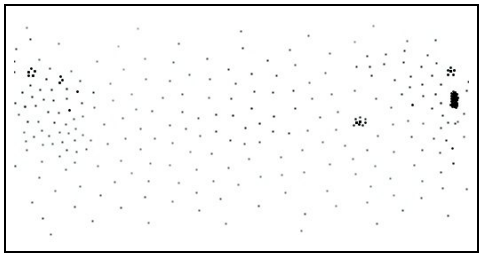


Figure 11-8. Ninja Shadow with Sixteen Samples per Pixel

[Bunnell & Pellacini, *GPU Gems*]

The Galileo map

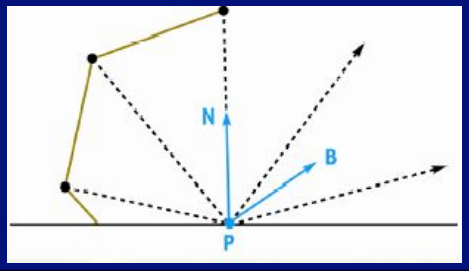


Structured importance sampling w/ 300 samples

[Agarwal, Ramamoorthi, Belongie, & Jensen 2003]

Ambient Occlusion: Main Idea

- At each point find
  - –Fraction of hemisphere that is occluded
  - –Visible fraction of hemisphere: (1-occlusion)
  - –And average unoccluded direction B
    - Use B for lighting (see later)

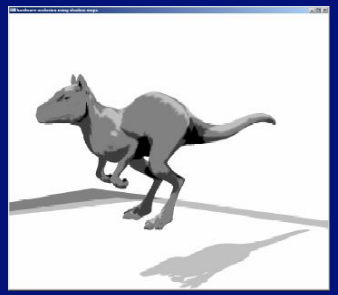


Computing the values: RT

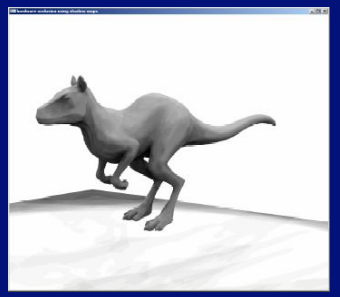
```
For each triangle {
    Compute center of triangle
    Generate rays over hemisphere
    Occlusion = 0
    For each ray
        If ray intersects objects   ++occlusion
    Occlusion /= nRays
}
```


Computing the values: SM

- Create shadow maps from N lights
- Check visibility of point wrt each light and determine occlusion: accumulation buffer



4 samples



32 samples

Computing the values: SM

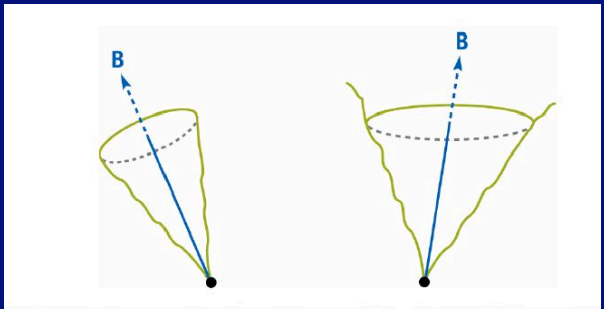
512 samples

Ambient occlusion: using the values

- Modulate diffuse shading
  - –Kd * (1-occlusion) * N.L
- Modulate irradiance map lookup

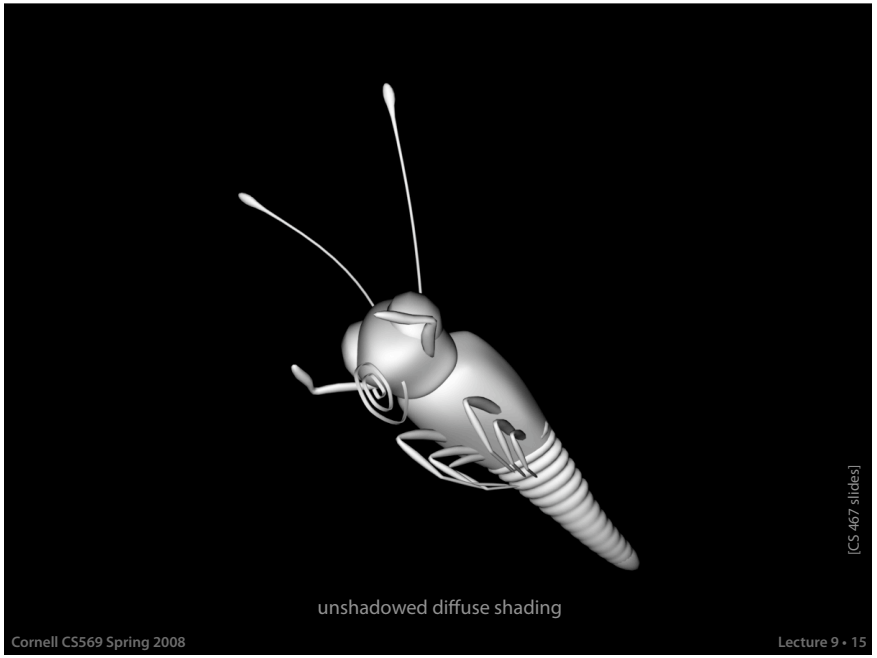
What about B?

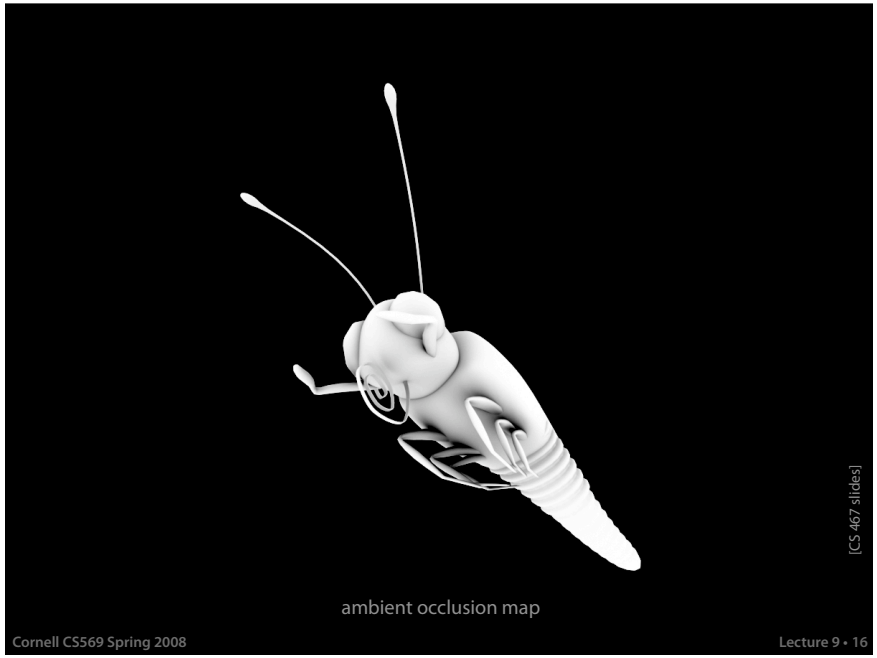
- The unoccluded direction gives an idea of where the main illumination is coming from
- This is called the “bent normal”

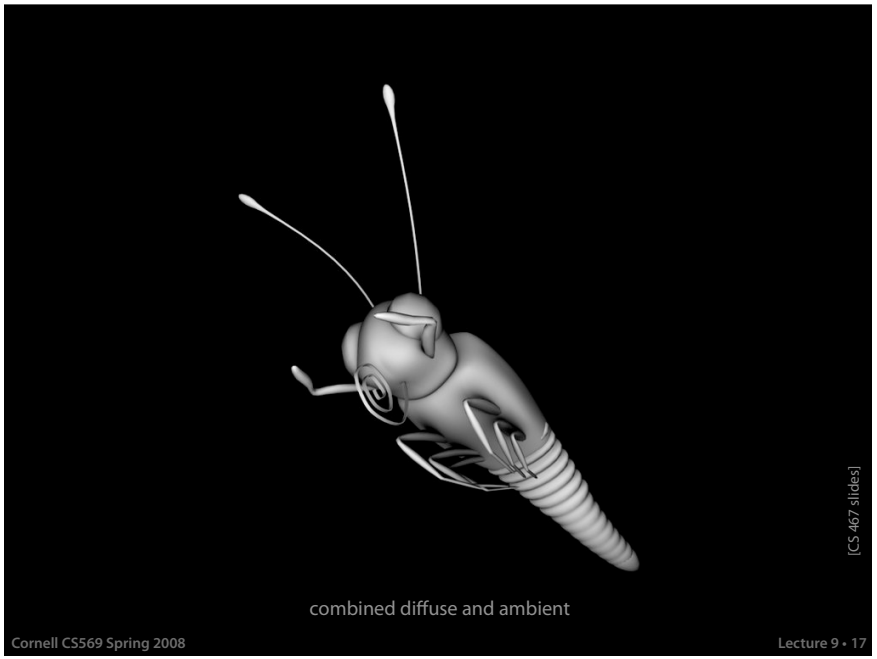


Computing the values: RT

```
For each triangle {
    Compute center of triangle
    Generate rays over hemisphere
    Occlusion = 0
    Avg dir = (0,0,0)
    For each ray
        If ray intersects objects   ++occlusion
        Else avg dir += ray.dir
    Occlusion /= nRays
    Normalize (avg dir)
}
```





[CS 467 slides]

combined diffuse and ambient